НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

до проекту

із дисципліни «Операційні системи»

на тему

Створення microframework WinUI на базі WinAPI

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-62 | асистент Громова В. В. |
| Антонов М. О. |  |

Київ — 2019

Зміст

[ВСТУП 2](#_Toc4864782)

[1 МЕТА РОБОТИ 3](#_Toc4864783)

[1.1 Мета роботи 3](#_Toc4864784)

[1.2 Поставленні задачі 3](#_Toc4864785)

[2 ОСНОВНА ЧАСТИНА 4](#_Toc4864786)

[2.1 Створення циклу обробки повідомлень 4](#_Toc4864787)

[2.2 Створення класів UI елементів 4](#_Toc4864788)

[3 КЛАС APPLICATION 5](#_Toc4864789)

[3.1 Public методи класу Application 5](#_Toc4864790)

[3.2 Private поля класу Application 5](#_Toc4864791)

# ВСТУП

Звіт створений для ознайомлення з процесом створення microframework-у на основі WinAPI. Проект включає в себе створення бібліотеки для побудови графічного інтерфейсу та реалізації функціоналу Windows WinAPI. Також створені програмні засоби на основі створеної бібліотеки WinUI.

У цьому звіті також описаний функціонал саме бібліотеки WinUI. Усі створені класи бібліотеки та їх зовнішні інтерфейси. Окрім цього описана внутрішня реалізація класів бібліотеки.

# 1 МЕТА РОБОТИ

### 1.1 Мета роботи

- Створити Framework для зручного використання функціоналу WinAPI.

### 1.2 Поставленні задачі

- Створити ієрархію класів для реалізації об’єктної моделі представлення.

- Створити зручні інтерфейси класів для використання їх у побудові програм.

- Створити з класів статичну бібліотеку на мові С++ для можливості використання бібліотеки.

# 2 ОСНОВНА ЧАСТИНА

### 2.1 Створення циклу обробки повідомлень

Для створення програми використовуючи WinAPI, в першу чергу треба створити цикл обробки повідомлень, які посилає операційна система із черги. Для вирішення цієї задачі було створено клас *Application*, який запускає цикл обробки повідомлень та отримує *HINSTANCE* програми.

### 2.2 Створення класів UI елементів

# 3 КЛАС APPLICATION

### 3.1 Public методи класу Application

*Application()* – конструктор класу проводить ініціалізацію поля класу *m\_haccel*, яка призначена для збереження ацелелятору. Ініціалізація проводиться використовуючи функцію *LoadAccelerators*.

*void exec()* – функція створює цикл *while*, який обробляє повідомлення з операційної системи. У циклі зчитується повідомлення типу *MSG,* використовуючи функцію *GetMessage*. Після зчитування повідомлення, воно обробляється та прибирається з черги повідомлень функціями *TranslateMessage* та *DispatchMessage*.

*void quit()* – надає можливість вийти з програми на будь-якому етапі виконання, за допомогою функції *PostQuitMessage*. Функція є статичною тому може бути викладена у будь-якому місці програми.

*static HINSTANCE getInstance()* – статична функція, повертає HINTANCE програми.

### 3.2 Private поля класу Application

*static HINSTANCE instance* – статичне поле класу, яке зберігає *HINSTANCE* програми. Поле класу є статичним, тому до нього може бути наданий доступ з будь якого місця програми, за допомогою функції *getInstance*.

*HACCEL m\_haccel* – Поле типу *HACCEL*. Ініціалізація поля проводиться у конструкторі та зберігає воно дані про ацелелятор. Доступу до цього поля немає.

*MSG m\_msg* – використовується для зберігання повідомлення під час використання функції *GetMessage* у циклі обробки повідомлень.